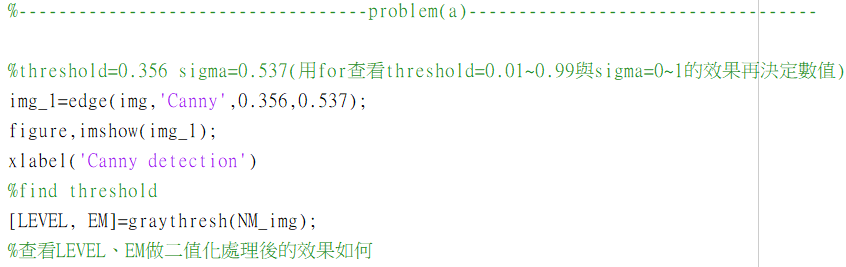
HW7 B063012054

(a)



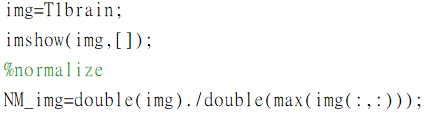


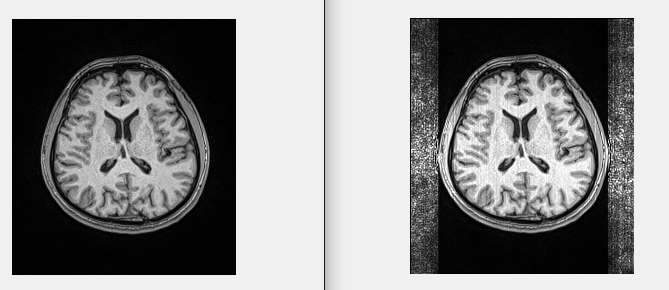
(b)

一開始在寫這題時因為直接使用graythresh(NM\_img)得到的值是永遠的0，查閱histogram發現分布圖長這樣：

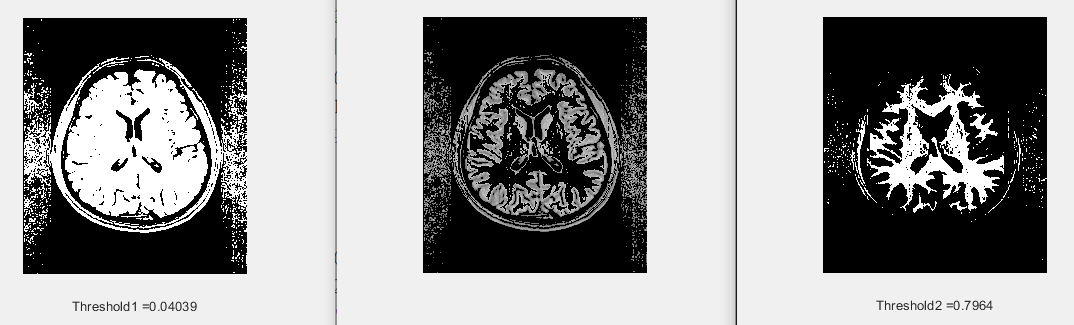


由於強度低的部分占比非常非常高，所以0是非常有可能的數值，試著將數值做正規化尋找閥值



結果很幸運的找出了兩個閥值，找到的圖非常漂亮，但奇怪的是左右邊有白條：

左圖為原圖 右圖為經過正規化處理的圖



上面三張由左而右分別為經過T1、T1<x<T2、T2處理後的圖，很明顯中間與右圖分離出了灰值與白質，但就是出現了白條，試著用閥值去除這些白點，例如設定經過正規化後先過濾一些點，其強度<T與NaN的地方設為0，取T=0.2但這會使得灰白質判斷劣化，且白點不會完全去除。因為這些白點的數值並不完全是0，其原本的數值可能是一個小值，通常分佈在強度0~200之間，因此單純用閥值概念過濾效果非常差。

回頭檢查code沒發現甚麼問題會造成白條，經過漫長的逐條檢查與維度檢查發現

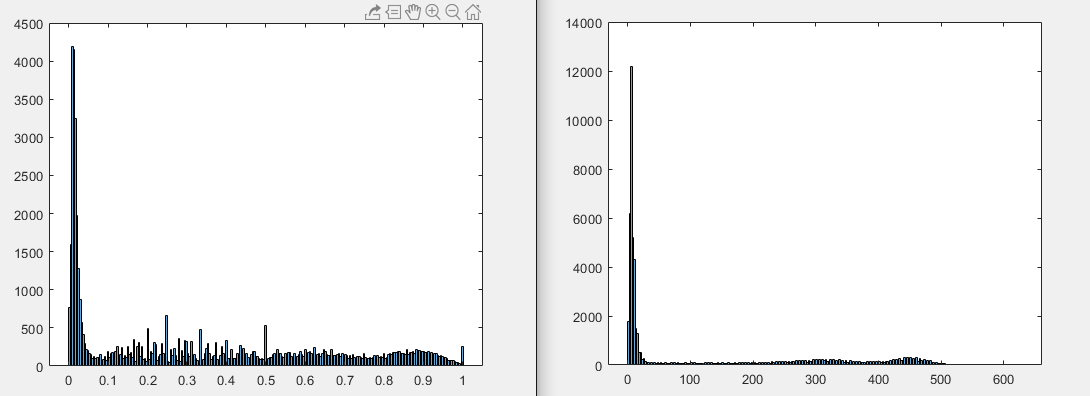
NM\_img=double(img)./double(max(img(:,:)));

有很大的問題，原本打算找尋整個圖片最強的地方，即最大值，因此使用max(img(:,:))

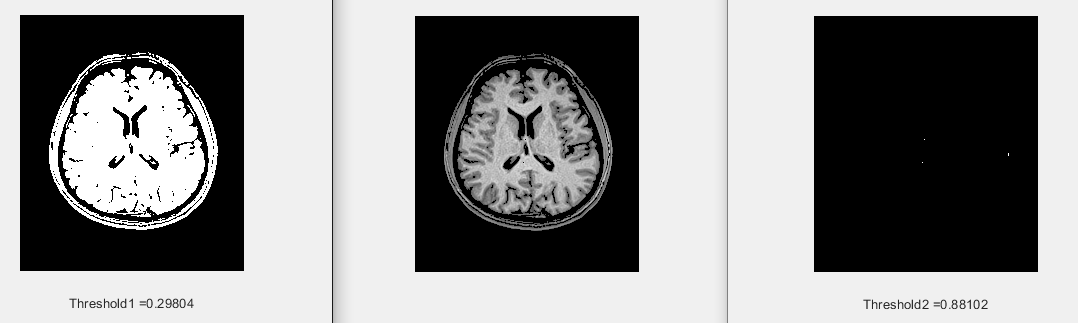
但此用法是找出每一個column的極大值，其返回的dim=(1,224)，應該要是(1,1)才對…

改成NM\_img=double(img)./double(max(img(:)));

理論上這樣才是正確的正規化，但發現圖片就一團亂了。



觀察一下兩者的histogram，左圖為誤用max，右圖為應正確使用max，之所以能正確辨識出灰白質其分佈應該要盡可能有”一坨一坨且不會極端分佈於某一坨”才適合找分界的閥值，理想是波谷要大且明顯、波峰不能太高，右圖就是非常差的例子，極端分佈於某一坨。



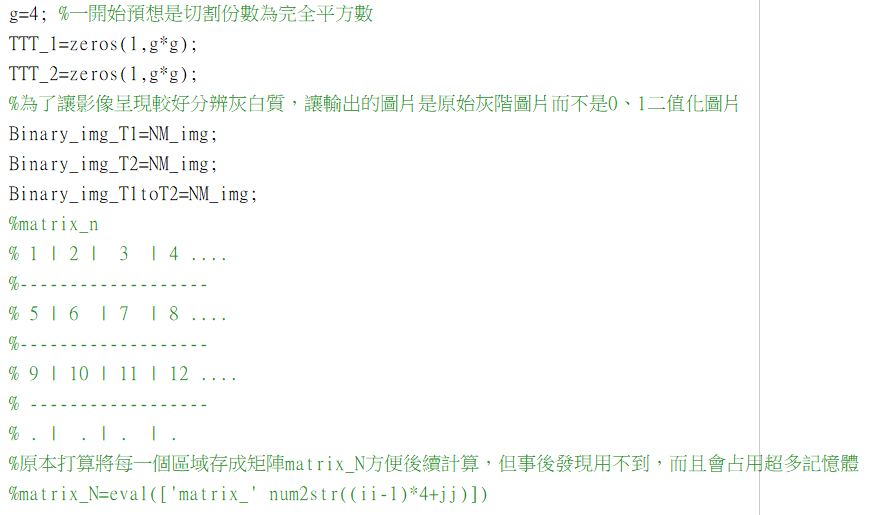
因此該題如果直接使用graythresh理論上無法有結果，會如上三圖所示，若是人工設計optimum global threshold，很明顯看到在histogram中0~100是一坨、200~350是一坨，最後是400~500這坨，因此可以定兩個Threshold，其效果理論上會很不錯，接近誤用max所得的圖。

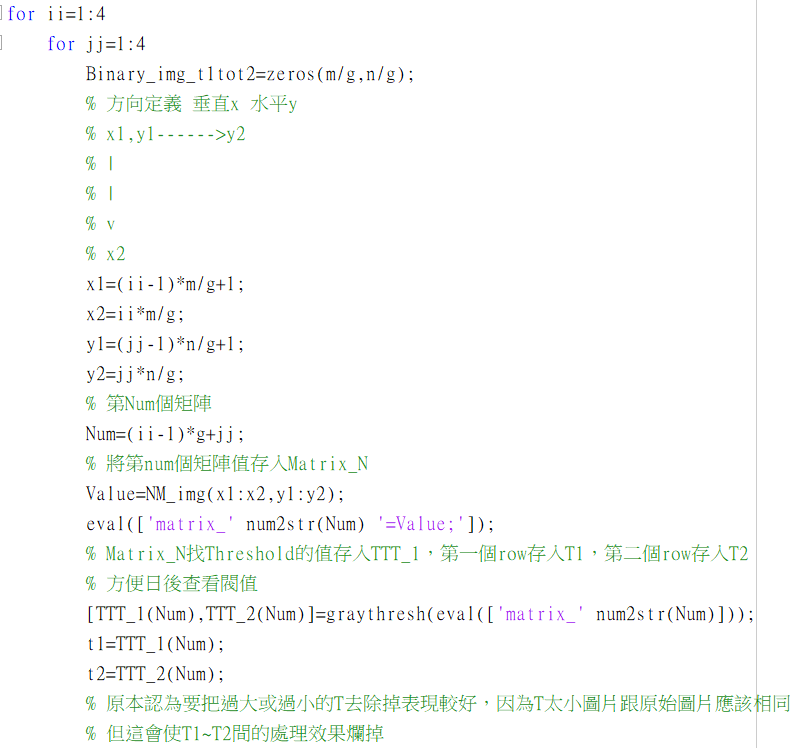
實作一下：人工選定LEVEL=T1=150、EM=T2=350，再做正規化，效果確實不錯

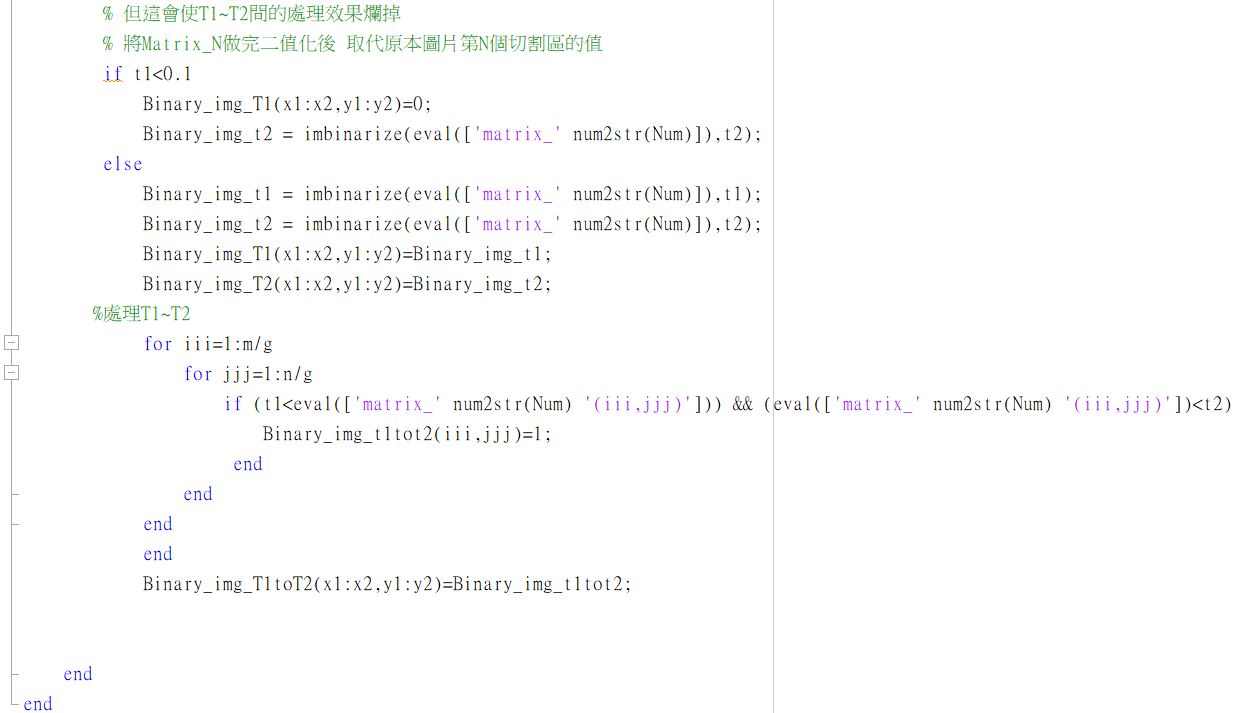
(T1=150/629) (T2=350/629)

(c)第一個想法：觀察原本的histogram，用肉眼使用(a)小題的edge detection將邊界銳利化，效果如講義p.10，但一定不可行，因為講義裡的圖幾乎都是用兩種顏色灰、白的圖，而非黑灰白，因此用edge detection效果會很好，但作業的圖有黑灰白，無法直接用edge detection再套用global thershold分出。

第二個想法：使用Image partition，第一次將影像分割成4\*4個區域，各自找尋T1與T2，效果如下：









T1 T2 T1~T2

中間圖片(T2)表示白質，因為>T2的地方都是比較白的地方，符合想法

但T1~T2的地方應該要是灰值，看起來是有抓到……但四周卻沒抓到

可能是以下原因



5

4

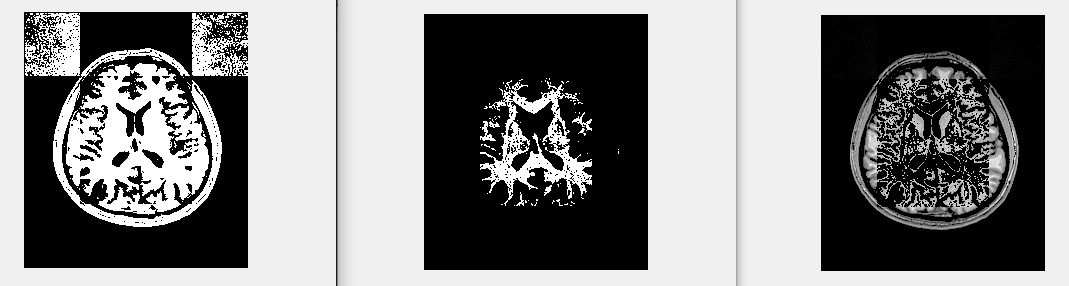
1

因為四周編號1.2.3.4.5.8.9.12.13.14.15.16的區域黑色成分(強度=0)太多，因此Threshold非常小，在處理的過程中使用T1<0.1時該區域二值化值全設為0，即不處理這個部分，原圖輸出，事後發現T1<0.1是多此一舉，就算不設定過濾值也會因為T1過小，呈現結果相同。當T1太小，輸出就是原圖，在二值化時該處全為1，因此在T1~T2處理時\*，呈現出來的圖就會使得周遭皆為原圖，只有在中間區域編號6.7.10.11處因符合T1~T2會進行過濾，二值化輸出，因此會將1的地方顯示出來。

\*之所以在T1~T2這張圖二值化輸出為0保持原圖、輸出為1則輸出二值化的圖是因為T1~T2並不是使用imbinarize，而是自寫for迴圈判斷，只有在Matrix\_N裡的值符合T1<pixel<T2時才會將””這個區域””的二值化值放入Binary\_img\_T1toT2(這是映出圖片的矩陣)，若不符合T1<pixel<T2則”這個區域”不做任何動作，因此映出的圖片會是當初最前面賦予的值。

由於寫得很亂，腦袋當機，隔天將切割份數加大，上課時有提過是計算一個3\*3或5\*5的kernel裡的平均值再找T，因此若將每一個格子做得愈小，效果應該要比較好才對，將參數整理優化，並調整輸出圖片呈現方式方便debug與閱讀，T2呈現的圖片使用二值化圖片，T1~T2輸出的圖片為”灰質

”區域圖片：



T1 T2 T1~T2

上面三張是換個寫法的4\*4切割結果呈現，四周的問題依然存在。



上面三張是換個寫法的16\*16切割結果呈現，好像更糟了，灰質部分參雜了大量白質的地方

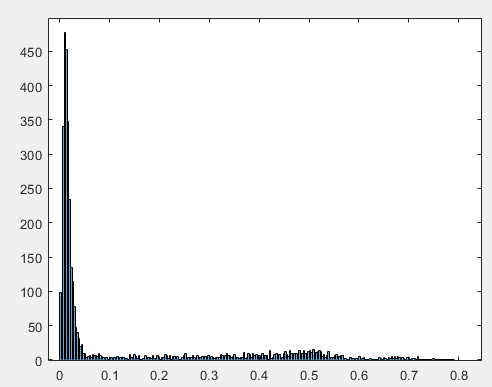


上面三張是32\*28切割(每一個kernel=8\*8)結果呈現，灰質部分依然參雜了應該要是白質的地方



上面三張是換個寫法的64\*56切割(每一個kernel=4\*4)結果呈現，雖然看起來很奇異，但仔細一看

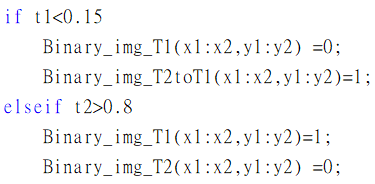
右圖確實是呈現了灰質區域，只是因為精準度要求高(kernel很小)，因此長得很鋸齒，白質部分長得很詭異，但可能也是精準度高的關係。



將4\*4切割時，編號5的區域畫出histogram尋找問題解法

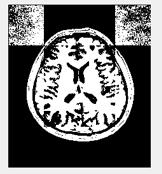
其對應T1=0.1961、T2=0.9001，觀察histogram可以清楚知道T1是可行的，但T2….非常有問題

因此解決方可能是：若T1、T2太高或太低，例如黑色佔多的格子T1會非常小(=0.0039)，白色佔多的格子T2會非常大(=0.8710)，那就只取一個適當T當閥值，例如：



但這樣子做毫無意義，因為這就變成人工去找尋閥值，而不是函數找到的，經過一整天的改良，這個方法確實無法完美分出灰白質。歸納以下

1. 假設T1=0.4以下為黑色、T2=0.7以上為白色，0.4~0.7為灰色
2. 在全黑部分，T1會非常小，但是T2不一定會適合當閥值，例如在4\*4切割時，編號1的區域T1=0.0078、T2=0.5905直接套用imbinarize會將區域分成

由於有些點強度>0.0078但看起來是黑色的部分會被分錯區域，因此不設定T1的閥值根本無法正確分類。

1. 同樣的，在全白部分，T2會非常大，因此在影像呈現時白質與灰質因為影像強度差不多，在分類時很容易會混進去，例如在32\*28與16\*16切割時灰白質混在一起。
2. 由2、3點可知道切割的大小影響判斷非常明顯，若一個kernel分割過小或過大，相似的對象太多，很容易因為雙閥值而混在一起，理論上kernel愈小精細度愈高分辨率也會愈高，但由於必須比較周遭的強度值，過少的比較對象會使說服率下降。
3. 為了解決雙閥值問題，曾想過若T2>0.8則T2=(T1+T2)/2或T2由T1替代剩下單閥值之類的想法，但這也等同於人工干擾加油添醋，既然要自行決定閥值，那不如一開始就由原圖的histogram人工審核訂定T1與T2。
4. 對於graythresh沒有太多的資料，其輸出為[LEVEL EM]，但EM卻沒有解釋得很清楚，究竟是不是代表T2，觀察切割完的每一區塊的histogram，其EM有時看起來就不是T2，而是整個分佈的上限值，但有時看起來確實就是給予T2，或許是找不到T2時則顯示該資料的上限。
5. 這個方法，使用graythresh，不適合分兩群以上

第三個想法：K-mean cluster + region growing的概念，應該是最容易實現的，因為這是一題影像內容物分類的問題，而且是依照影像強度分類，首先在看起來黑色、白色、灰色的點當標的，計算該點與周圍的強度是否低於閥值，是的就屬於同類，再計算兩者的距離，互相比較誰近，做分類，即可完成，最困難的點應該在於如何決定灰色值是多少? 500? 550?不同的資料型態有不同的定義，為求方便，使用imsegkmeans這個內建函數分類，非常快速且正確地分出三個部分黑色、灰質、白質。

